Relationalizing RDF Stores for Tools Reusability

Sunitha Ramanujam[1], Anubha Gupta[1], Latifur Khan[1], Steven Seida[2], Bhavani Thuraisingham[1]
[1] The University of Texas at Dallas
800, West Campbell Road
Richardson, TX 75080-3021
{sxr063200, axg089100, lkhan, bxt043000}@utdallas.edu

[2] Raytheon Corporation
1200 South Jupiter Road
Garland, TX 75042
steven_b_seida@raytheon.com

ABSTRACT
The emergence of Semantic Web technologies and standards such as Resource Description Framework (RDF) has introduced novel data storage models such as the RDF Graph Model. In this paper, we present a research effort called R2D, which attempts to bridge the gap between RDF and RDBMS concepts by presenting a relational view of RDF data stores. Thus, R2D is essentially a relational wrapper around RDF stores that aims to make the variety of stable relational tools that are currently in the market available to RDF stores without data duplication and synchronization issues.

Categories and Subject Descriptors
D.2.12 [Interoperability]: Data Mapping and Interoperability between applications

General Terms
Algorithms, Design, Management.

1. INTRODUCTION
Every new data storage paradigm comes with its own demands for data modeling and visualization tools to simplify data management. In order to salvage the time, effort, and resources exhausted in the development of such tools for existing, mature technologies such as relational database management systems, it would be prudent to focus on research efforts that attempt to recycle these tools for new data models such as the RDF Graph Model. R2D is one such effort that attempts to eliminate the learning curves associated with mastering new tools and to leverage the advantages offered by the relational tools while continuing to reap the benefits provided by the newer web technologies and standards such as RDF.

R2D, which could be considered as the complement of D2RQ [1] since it works in the reverse direction, uses a declarative mapping scheme for the translation of RDF Graph structures to equivalent relational schema constructs. Functionalities provided by R2D are:

- Ability to infer the entities comprising the RDF store, their attributes, and the relationships that exist between them.
- Ability to generate a meaningful, normalized, domain-specific relational schema corresponding to the RDF store.
- Ability to appropriately transform blank nodes, RDF containers, and RDF collections to relational entities/attributes.
- Ability to access information in a non-RDBMS data store using relational data visualization tools
- Ability to query a non-RDBMS data store using conventional SQL statements

R2D is implemented as a JDBC wrapper around RDF stores and the system architecture is illustrated in Figure 1.

WWW 2009, April 20-24, 2009, Madrid, Spain.
ACM 978-1-60558-487-4/09/04.

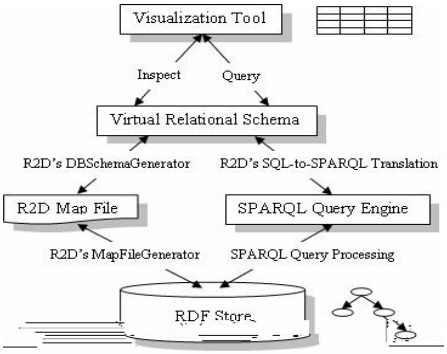

Figure 1. R2D System Architecture

At the heart of the transformation of RDF Graphs to virtual relational database schemas is the R2D mapping language and details of the same are presented below.

2. R2D Mapping Constructs
The chief construct of the R2D mapping language is the TableMap, which refers to a table in a relational database. Each rdfs:class object in the RDF store maps to a distinct r2d:TableMap, and, in the absence of rdfs:class objects, the r2d:TableMaps are inferred from the instance data in the RDF Store. Each TableMap entity has a set of columns which correspond to the predicates associated with the resource mapped by the TableMap. Simple predicates are mapped using the r2d:ColumnBridge construct while multi-valued predicates are mapped using the r2d:MultiValuedColumnBridge construct. Foreign key relationships are handled using the r2d:refersToTableMap construct. R2D also supports a variety of blank node scenarios such as single or multi-valued literal blank nodes, single or multi-valued resource blank nodes, and mixed (literal/resource) blank nodes using a variety of constructs that are described in [2]. RDF features such as RDF Containers and RDF Collections are also supported using the above-mentioned blank node mapping constructs. Lastly, R2D provides the r2d:MultiValuedPredicate construct to handle RDF triples that essentially map to multi-valued attributes in the relational domain. The "MultiValued" constructs are vital to ensure the generation of a normalized relational schema corresponding to the RDF store.

3. RDF Modules
There are three modules comprising the R2D framework. A) The first module is the RDFMapFileGenerator module which takes an RDF triples database as input and produces a mapping file as output. The map file generator includes detailed specifications to handle a variety of RDF blank nodes, containers, and collections. This module can be bypassed in the presence of a domain expert who can provide the mapping file manually. B) The second module is the DBSchemaGenerator, which parses the map file generated by the

first module and, for the RDF store, presents a list of relational tables, columns, and the relationships between them. C) The last R2D module is the SQL-to-SPARQL transformation process that transparently converts any SQL statement issued against the virtual relational schema generated by DBSchemaGenerator into its SPARQL equivalent, and transforms the results obtained from the SPARQL Query Engine into a relational/tabular format. This module includes the ability to translate SQL pattern matching and aggregation functionality into appropriate SPARQL clauses where applicable.

4. SAMPLE SCENARIO

The example RDF database stored using Jena 2.5.6 and considered for experimentation and elucidation purposes is one which has information pertaining to employees, departments, and projects. Employees have literal properties such as Name and Address. They also have blank nodes that contain phone numbers and projects information for the employee, and a resource predicate that associates a department with each employee. The map file excerpt corresponding to the Employee entity is listed in Figure 2 along with the relational schema corresponding to the RDF store, as seen through the relational visualization tool, DataVision [3]. A more detailed description of the various R2D Mapping constructs and schema generation processes can be found in [2].

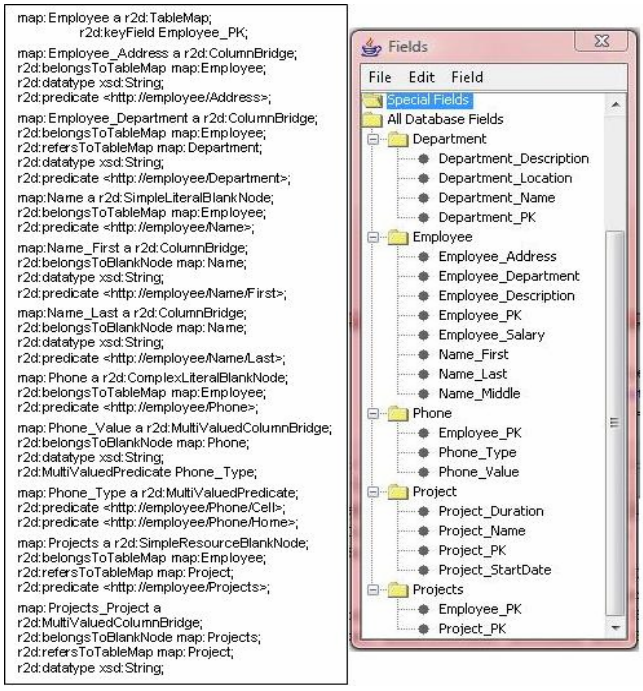

Figure 2: Map File Excerpt for "Employee" and Relational Schema for RDF Store

The mapping information from the map file is used by the SQL-to-SPARQL translator to convert any SQL statements issued against the virtual relational schema into its SPARQL equivalent. One such query issued through DataVision and its converted SPARQL equivalent is illustrated in Figure 3.

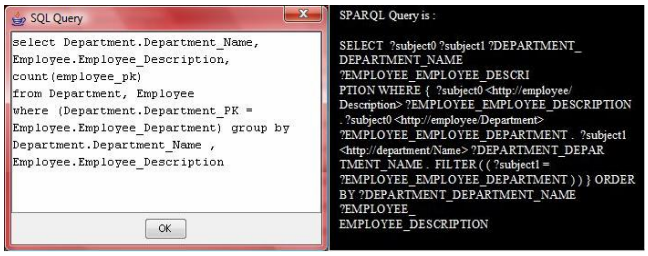

Figure 3: SQL-to-SPARQL Transformation

5. PERFORMANCE RESULTS

The performance of data retrieval using R2D was compared with that obtained using an RDF visualization tool called GRUFF [4] using a variety of queries of varying complexity. R2D queries were fired against Jena's in-memory data store. Table 1 lists the results obtained for an RDF triples database size of 0.5M.

Table 1: Query Performance Results

QUERY	GRUFF	R2D
3 Projections, 1 Where clause for LIKE, 2-table join	315secs	8secs
4 Projections involving properties and SimpleLiteralBlankNodes, 3 Where clauses involving LIKE and equality operators connected using conjunction and disjunction, 2-table join	315secs	6secs
3 Projections involving properties and SimpleResourceBlankNodes, 3-table join	600secs	6secs
4 Projections from both types of blank nodes above, aggregation function using Group By, 3-table join	550secs	8secs

The performance results highlight the fact that R2D's performance is far superior to that of the direct RDF visualization tool. Further, the time taken by the SQL-to-SPARQL translation process is negligible and, hence, R2D does not add any overheads to the SPARQL Query processing performance.

6. CONCLUSION

In today's highly web-enabled world, R2D offers users the ability to reuse existing knowledge and resources by enabling the integration of traditional mature and stable relational tools with the newer semantic web technologies such as RDF stores. The competitive query performance results obtained through R2D make it a viable contender in the RDF data visualization and management arena. Future work includes support for reification.

7. REFERENCES

[1] Bizer, C., and Cyganiak, R. D2R – Publishing Relational Databases on the Semantic Web. 5th International Semantic Web Conference, 2006

[2] Ramanujam, S., Gupta, A., Khan, L., Seida, S., Thuraisingham, B. A Framework for the Relational Transformation of RDF Data. UTD Technical Report UTDSC-40-08. http://www.utdallas.edu/~sxr063200/Paper2.pdf, 2008.

[3] DataVision. The Open Source Report Writer. http://datavision.sourceforge.net/

[4] GRUFF. A Grapher-Based Triple-Store Browser for Allegrograph. http://agraph.franz.com/gruff/